CMPE300 - Analysis of Algorithms
Spring 2019
Assignment 2

Due 11:59am, May 20rd, 2019

General Instructions

This assignment aims at helping you grasp some important aspects of the course. Please give your answers to the following questions clearly and thoroughly. Do not just give the final answer but explain how you derived it. Please work on it on your own and write down your very own solutions independently.

For any further questions, please feel free to reach the TA at emrah.budur@boun.edu.tr.

Submission guidelines

1. There are two questions in this assignment. Write your answers such that the answer for each of two questions starts in a new page just as given in this PDF file. It will help us a lot and speed up the online grading process.

2. Please upload your answers as a pdf file to www.gradescope.com, while logging in with your boun.edu.tr email address, until 11:59am on May 20th, 2019. This tutorial video may be of your help while submitting your assignment to Gradescope. *There is no need to hand in your answers as an hard-copy for this assignment.*

3. Note that the deadline is strict and late submissions will be affected by the late date policy of the course.

4. Every submission must be your own work and any similarities among the other submissions will be considered as cheating.

1 [40 points] Question 1

Suppose that you are given an array of size n where the values of the elements in the array are in a fixed range of integers between $[1 - k]$. An example of such array may be the ages of some people.

Find the lower bound, $\Omega(n)$, of sorting this kind of array in the worst case. Explain your reasoning clearly and thoroughly.

2 [60 points] Question 2

2.1 Motivation

Suppose that you have seen one of the most interesting flowers in the world that is shown in Figure 1, while you were hiking during the spring break.

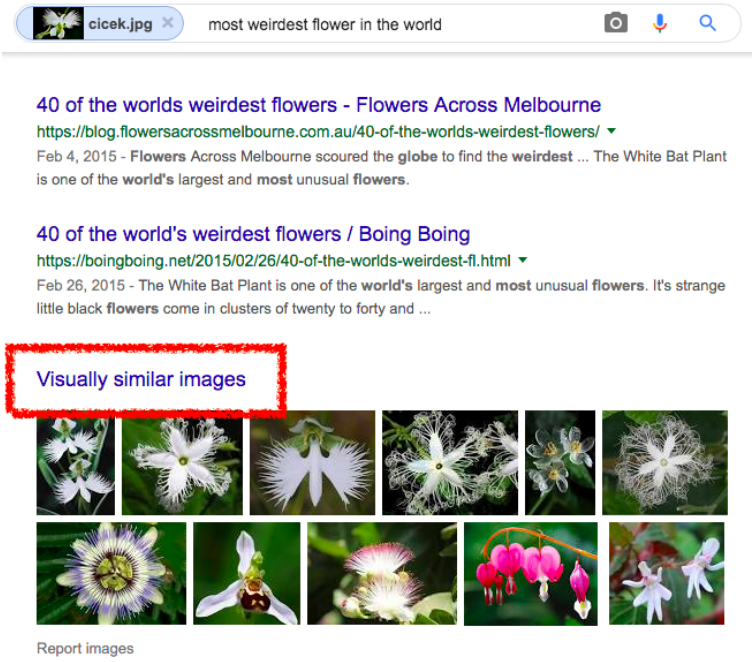
Figure 1: The interesting flower. You can download it from the link http://bit.ly/2PTp2QP

As an extremely curious explorer, you wanted to learn more about this flower. But you do not know anything about this flower to query from a search engine. Fortunately, the typical search engines provide a special service that allows you to query by raw images instead of keywords. An example of such search engine is https://www.google.com/images. By using this service, you got very happy to find lots of similar images by following the steps in Figure 2.

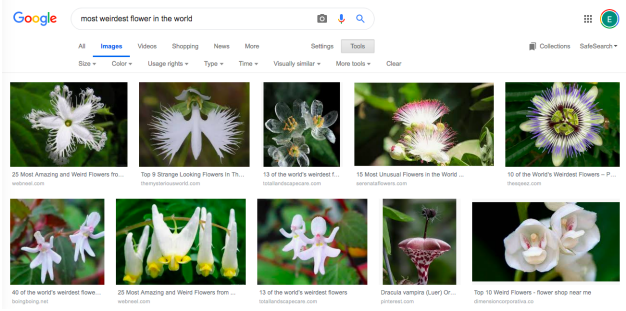
2.2 Problem Formulation

Let's now simplify and formulate this problem. Assume that you have an image of size $m \times m$ and also assume that the search engine has n images, all having the same size $m \times m$ in its entire index. Now, you want to find **exactly the same image** you have, from this kind of image search service. The naive approach for this problem would be to compare the pixels of the the query image against the pixels of all images in its index, in an elementwise way. In other words, the pixel value in $i'th$ row and $j'th$ column of the query image would be expected to be the same as the corresponding pixel value of the candidate image in the index, for all $1 \leq i, j \leq m$, so that the candidate image is listed in the resulting page. The worst case complexity of this naive approach would be $W(m, n) \in O(m^2 n)$.

(a) Click on the *Search by image* button.



(b) Click on the *Visually similar images*.



(c) Browse the list of similar images.

Figure 2: Finding new images that are similar to the image you want to explore.

2.3 The Main Problem

The main problem is to figure out whether two given images having the same size $m \times m$ are identical or not. The naive approach of this main problem has the worst case complexity $W(m) \in O(m^2)$ due to elementwise comparison of two $m \times m$ matrices.

- **(30 points)** Develop a false-biased Monte-Carlo algorithm to solve **the main problem** with a better worst case complexity, $W'(m)$, such that $W'(m) \in \textit{Little-o}(m^2)$.
- **(20 points)** Calculate the p value of your algorithm with a clear reasoning. The p-value of your algorithm may also depend on the parameter(s), if any, in your algorithm.
- **(10 points)** Write the pseudocode of your algorithm.